Worked solution - Tutorial Sheet 5

Question 1

```
restart;read "C:/JF/Software/Maple/Start.mpl";Digits:=6:with(plots):
```

$$GVFE := \frac{S - \frac{Q^2}{K^2}}{1 - \frac{\beta Q^2 B}{g A^3}};$$

$$GVFE := \frac{S - \frac{Q^2}{K^2}}{1 - \frac{\beta Q^2 B}{g A^3}} \qquad \textbf{(1.1)}$$

```
B:=W+2*m*h;
A:=h*(W+m*h);
P:=W+2*sqrt(1+m^2)*h;
K:=1/n*A^(5/3)/P^(2/3);
```

$$B := W + 2\,m\,h$$

$$A := h\,(W + m\,h)$$

$$P := W + 2\sqrt{1+m^2}\,h$$

$$K := \frac{(h\,(W + m\,h))^{5/3}}{n\left(W + 2\sqrt{1+m^2}\,h\right)^{2/3}} \qquad \textbf{(1.2)}$$

Introduce the numbers given (we use the slightly inconsistent notation that $h[0]$ is the depth at $-\infty$ and $h[1]$ the depth at the barrage.

$S := 10.^{-4} : n := 0.02 : h[1] := 2.5 :$

$h[0] := 2 : W := 10 : m := 2 : \text{beta} := 1.05 : g := 9.8 :$

$L := 20000 :$

Calculate the discharge from the normal depth given:

```
Q:=subs(h=h[0],K*sqrt(S));Q:=evalf(Q);
```

$$Q := \frac{14.0000\,28^{2/3}}{\left(10+4\sqrt{5}\right)^{2/3}}$$

$$Q := 18.1654 \tag{1.3}$$

The right side of the differential equation is

GVFE;

$$\frac{0.000100000 - \frac{0.131993\left(10+2\sqrt{5}\,h\right)^{4/3}}{\left(h\,(10+2\,h)\right)^{10/3}}}{1 - \frac{35.3552\,(10+4\,h)}{h^3\,(10+2\,h)^3}} \tag{1.4}$$

Test the size of the inertia term

Froude number

F[0]:=subs(h=h[0],Q*sqrt(B/g/A^3));

Magnitude of inertia term: beta*F[0]^2;

$$F_0 := 0.166162$$

$$0.0289903 \tag{1.5}$$

We could easily leave it out, but we won't.

Initial conditions

$Z[1] := 0:$

$\delta := 2000:$

$x[1] := 0.:$

$imax := \frac{L}{delta}:$

Euler method

```
for i from 1 to imax do
  x[i+1] := x[i]-delta;
  h[i+1] := h[i]-delta*evalf(subs(h=h[i],GVFE)); The differential equation
  Z[i+1] := Z[i]+delta*S; Points on the bed
  printf("\n%2d %6.0f %6.3f",i+1, x[i+1], h[i+1]);
end do:
```

```
 2  -2000  2.386
 3  -4000  2.289
 4  -6000  2.209
 5  -8000  2.147
```

```
 6 -10000  2.100
 7 -12000  2.066
 8 -14000  2.043
 9 -16000  2.028
10 -18000  2.018
11 -20000  2.011
```

It has always surprised me how far a backwater curve extends for gentle slopes

Repeat with half the step size:

```
delta := (1/2)*(1*delta);
x2[1] := 0;
h2[1] := h[1];
Z2[1] := Z[1];
i2max := L/delta;
```

$$\delta := 1000$$

$$x2_1 := 0$$

$$h2_1 := 2.5$$

$$Z2_1 := 0$$

$$i2max := 20 \tag{1.6}$$

Euler method, halving the steps

```
for i from 1 to i2max do
   x2[i+1] := x2[i]-delta;
   h2[i+1] := h2[i]-delta*evalf(subs(h=h2[i],GVFE));
   Z2[i+1] := Z2[i]+delta*S;
   Uniform[i+1]:=h[0]-S*x2[i+1];
   printf("\n%2d %6.0f %6.3f",i+1, x2[i+1], h2[i+1]);
end do:
```

```
 2  -1000  2.443
 3  -2000  2.390
 4  -3000  2.341
 5  -4000  2.296
 6  -5000  2.256
 7  -6000  2.219
 8  -7000  2.187
 9  -8000  2.158
10  -9000  2.133
11 -10000  2.112
```

```
12 -11000  2.093
13 -12000  2.077
14 -13000  2.064
15 -14000  2.053
16 -15000  2.044
17 -16000  2.036
18 -17000  2.029
19 -18000  2.024
20 -19000  2.020
21 -20000  2.016
```

Samuels' approximate formula

```
dPdh:=evalf(diff(P,h));
gamma:=evalf(subs(h=h[0],S/(1-beta*F[0]^2)*(10./3.*B/A-4/3*dPdh/P)));
Samuels:=h[0]+(h[1]-h[0])*exp(gamma*X);
```

$$dPdh := 4.47214$$

$$\gamma := 0.000188269$$

$$Samuels := 2 + 0.5\, e^{0.000188269\, X} \tag{1.7}$$

Accurate numerical method (not asked for) - uses in-built software

```
xx:='xx':
diff(H(xx),xx)=subs(h=H(xx),GVFE):
Sol:=dsolve({%,H(0)=h[1]},numeric):
SS:=[]:TT:=[]:
for i from 1 to imax+1 do
xx:=x[i];
hh:=op([2,2],Sol(xx));
SS:=[op(SS),[xx,hh]];
TT:=[op(TT),[xx,hh-S*xx]];
od:
```

Richardson extrapolation and comparison with accurate solution

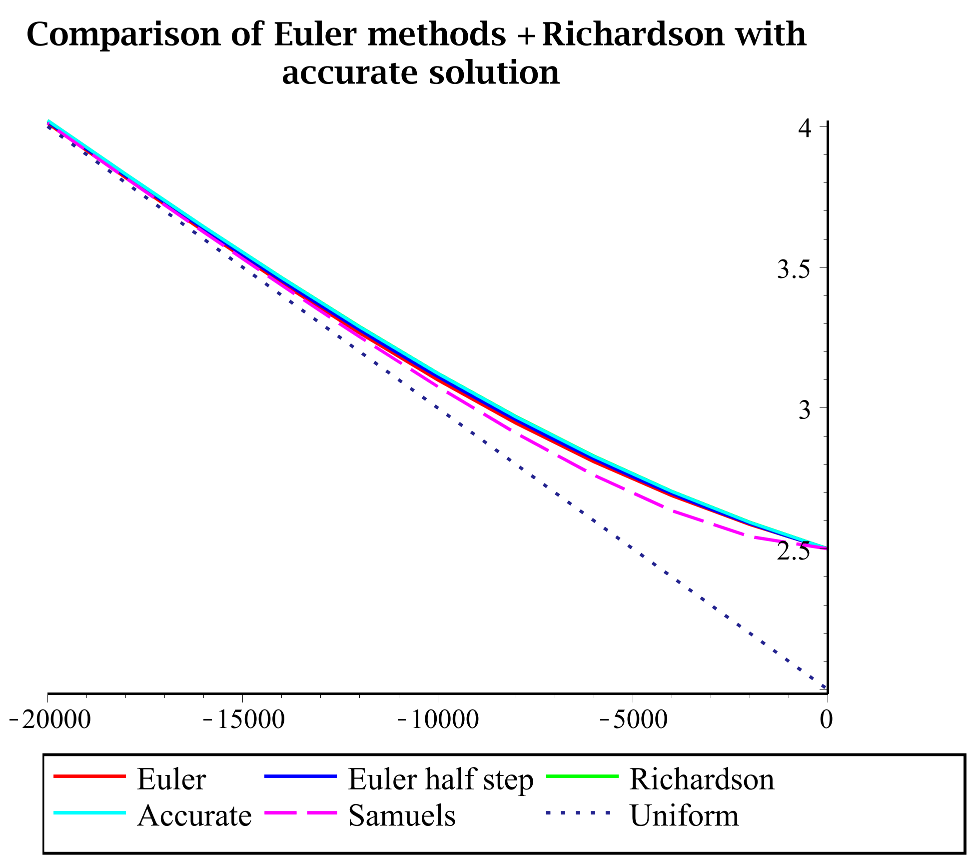
```
printf("        x  Euler 1 Euler 2 Richardson Accurate Samuels");
for i from 1 to imax+1 do
Rich[i]:=2*h2[2*i-1]-h[i];
Sam[i]:=subs(X=x[i],Samuels);
printf("\n%10.0f %6.3f %6.3f    %6.3f    %6.3f   %6.3f",x[i],h[i],h2[2*i-1],Rich[i],op(
[2,2],Sol(x[i])),Sam[i]);
od:
```

x	Euler 1	Euler 2	Richardson	Accurate	Samuels
0	2.500	2.500	2.500	2.500	2.500
-2000	2.386	2.390	2.394	2.394	2.343

-4000	2.289	2.296	2.304	2.303	2.235
-6000	2.209	2.219	2.229	2.229	2.162
-8000	2.147	2.158	2.170	2.169	2.111
-10000	2.100	2.112	2.124	2.123	2.076
-12000	2.066	2.077	2.089	2.088	2.052
-14000	2.043	2.053	2.063	2.062	2.036
-16000	2.028	2.036	2.044	2.044	2.025
-18000	2.018	2.024	2.030	2.030	2.017
-20000	2.011	2.016	2.021	2.021	2.012

Plot

```
plot([[[x[j],Z[j]+h[j]]$j=1..imax+1],[[x2[j],Z2[j]+h2[j]]$j=1..i2max+1],[[x[j],Z[j]+Rich[j]]$j=1..imax+1],TT,[[x[j],Z[j]+Sam[j]]$j=1..imax+1],[xxx,h[0]-S*xxx,xxx=-L..0]],title="Comparison of Euler methods +Richardson with accurate solution",color=[red,blue,green,cyan,magenta,navy],linestyle=[solid$4,dash,dot],titlefont=[Times,bold,12],legend=["Euler","Euler half step","Richardson","Accurate","Samuels","Uniform"]);
```



Our results, with Richardson extrapolation, are very accurate! However, any of the solution methods are good enough for practical purposes.
Samuels simple approximation is not so very accurate near the weir/obstacle - but does give a reasonable idea of the rate of decay upstream. If we had no other information, we could use this.

Question 2

Now solve the problem for slope 0.001 - but with smaller depths

```
restart;read "C:/JF/Software/Maple/Start.mpl";Digits:=6:with(plots):
```

$$GVFE := \frac{S - \frac{Q^2}{K^2}}{1 - \frac{\beta Q^2 B}{g A^3}} :$$

```
B:=W+2*m*h;
A:=h*(W+m*h);
P:=W+2*sqrt(1+m^2)*h;
K:=1/n*A^(5/3)/P^(2/3);
```

$$B := W + 2\, m\, h$$

$$A := h\, (W + m\, h)$$

$$P := W + 2\sqrt{1 + m^2}\, h$$

$$K := \frac{(h\,(W + m\,h))^{5/3}}{n\left(W + 2\sqrt{1 + m^2}\, h\right)^{2/3}} \qquad \textbf{(2.1)}$$

Introduce the numbers given:

$S := 10.^{-3} : n := 0.02 : h[1] := 1.5 :$
$h[0] := 1 : W := 10 : m := 2 : beta := 1.05 : g := 9.8 :$
$L := 1000 :$

```
Q:=subs(h=h[0],K*sqrt(S));Q:=evalf(Q);
```

$$Q := \frac{18.9737\; 12^{2/3}}{\left(10 + 2\sqrt{5}\right)^{2/3}}$$

$$Q := 16.7462 \qquad \textbf{(2.2)}$$

Check the magnitude of the inertia term:

Froude number `F[0]:=subs(h=h[0],Q*sqrt(B/g/A^3));`

$$F_0 := 0.481498 \tag{2.3}$$

Magnitude of inertia term: beta*F[0]^2;

$$0.243432 \tag{2.4}$$

Now we can't neglect it!

Initial conditionsw

```
δ := 100 :
x[1] := 0. :
Z[1] := 0. :
```

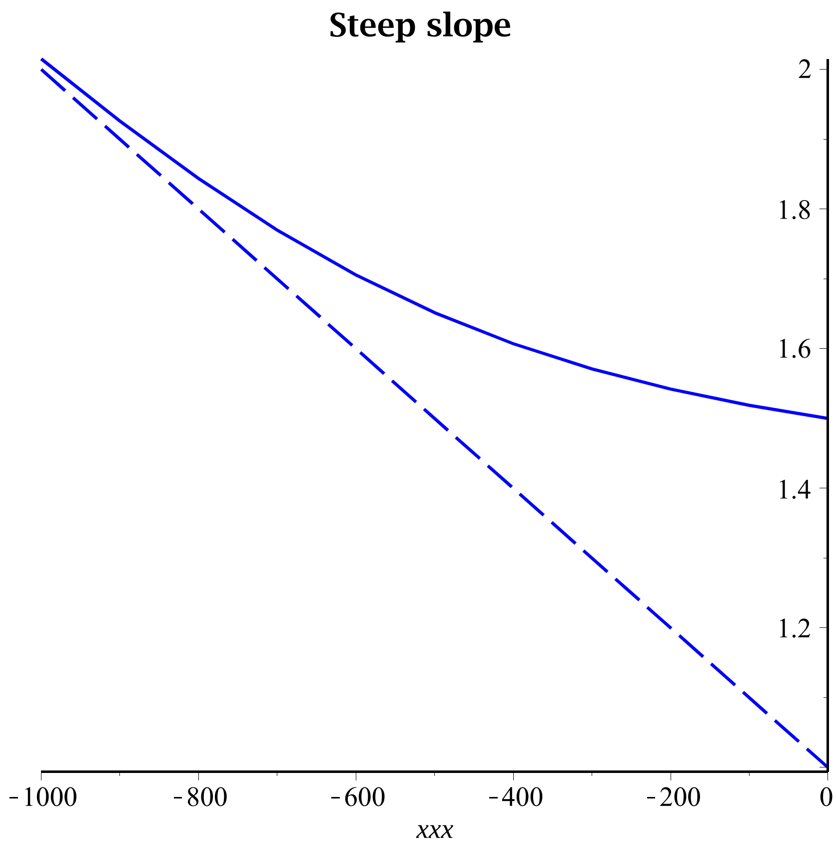
$imax := \frac{L}{delta}$:

Euler method

```
for i from 1 to imax do
  x[i+1] := x[i]-delta;
  h[i+1] := h[i]-delta*evalf(subs(h=h[i],GVFE)); The differential equation
  Z[i+1] := Z[i]+delta*S; Points on the bed
  printf("\n%2d %6.0f %6.3f",i+1, x[i+1], h[i+1]);
end do:
```

```
 2   -100  1.419
 3   -200  1.342
 4   -300  1.271
 5   -400  1.207
 6   -500  1.152
 7   -600  1.106
 8   -700  1.070
 9   -800  1.044
10   -900  1.026
11  -1000  1.015
```

```
SSS:=[[x[j],h[j]+Z[j]]$j=1..imax+1]:
plot([SSS,h[0]-xxx*S],xxx=-L..0,title="Steep slope",legend=["",""],color=[blue],
linestyle=[solid,dash],titlefont=[Times,bold,12]);
```



Now the length of the backwater curve is only about 1km, compared with 20km for the gentle slope case.

In lectures we had the approximate formula for the decay constant

$\gamma = \frac{10}{3} \frac{S}{h[0]}$: We see then that after a distance upstream $x = \frac{h[0]}{S}$: the curve will decay to a value of exp(-10/3.)=0.035: namely 3.5% of the backed up value. This is not a bad estimate of the "length" of the backwater curve: In our first example this was 2/10^(-4) = 20000: and in the second, 1/10^(-3)=1000, both of which are roughly right.